International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 12 | Dec -2016 www.irjet.net p-ISSN: 2395-0072

GENERATING SECRET KEY FOR MULTI-KEYWORD RANKED SEARCH OVER ENCRYPTED CLOUD DATA

P. Anitha[1], D.Vijayalakshmi[2]

[1] *Assistant Professor, Dept. of Computer Applications, Vellalar College for Women, Erode, Tamilnadu, India*
[2]*Research Scholar, Department of Computer Science, Vellalar College for Women, Erode, Tamilnadu, India*

Abstract - *The Cloud Computing, it provide more security for the data owners and clients. As increasing popularity of cloud computing, more and more data owners are motivated to their data to cloud servers for great convenience and reduced cost in data management. So, sensitive data should be encrypted before outsourcing for privacy requirements, which obsoletes data utilization like keyword- based document retrieval. A secret key generation for multi-keyword ranked search over encrypted cloud data, which simultaneously dynamic update operations like modify and view of documents. Specifically, the vector space model and the widely-used TF×IDF model are combined in the index construction and query generation. The secure kNN algorithm is utilized to encrypt the index and query vectors, and mean while ensure correct relevance score calculation between encrypted index and query vectors. In order to provide security, the secrete key is generated for each data owners individual electronic mail. That secrete key can be copied and pasted for further login. When the user wants to change their password the secret key also been updated. The user's privacy more increased. Due to the use of secrete key security is highly increased, While outsourcing the data. User can search and find the document using search command that is displayed in Rank based.*

Key Words: Multi-keyword ranked search, dynamic update, dynamic secret key, privacy preserving, cloud computing

1. INTRODUCTION

Cloud computing is the extended dreamed vision of computing as a utility, where cloud customers can remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable Its great flexibility and economic savings are motivating both individuals and enterprise to outsource their local hard data management system into the cloud. To defend data privacy and combat unwanted accesses in the cloud and outside, sensitive data, e.g., emails, personal health records, tax documents, financial transactions, etc., may have to be encrypted by data owners before outsourcing to the marketable public cloud. this, however, obsoletes the traditional data utilization service based on plaintext keyword search. The small solution of downloading all the data and decrypting locally is clearly impractical, due to the huge amount of bandwidth cost in cloud scale systems. These methods are not practical due to their high computational slide for both the cloud sever and user. On the contrary, more practical special purpose solutions, such as searchable encryption (SE) schemes have made specific assistance in terms of efficiency, functionality and security. Searchable encryption schemes enable the client to store the encrypted data to the cloud and execute keyword search over cipher text domain.

They secure tree-based search scheme over the encrypted cloud data, which supports multi-keyword ranked search and dynamic operation on the document gathering. Specifically, the vector space model and the widely-used "term frequency (TF) × inverse document frequency (IDF)" model are collective in the index structure and query generation to provide multi-keyword ranked search. In order to gain high search efficiency, they construct a tree-based index structure and propose a "Greedy Depth-first Search" algorithm based on this index tree.

2. LITERATURE REVIEW

Zhihua Xia, Xinhui Wang, Xingming Sun and Qian Wang, [10] proposal describes sensitive data should be encrypted before outsourcing for privacy requirements, which obsoletes data utilization like keyword-based document retrieval. In this paper, we present a secure multi-keyword ranked search scheme over encrypted cloud data, which simultaneously supports dynamic update operations like deletion and insertion of documents. Specifically, the vector space model and the widely-used TF×IDF model are combined in the index construction and query generation. The secure kNN algorithm is utilized to encrypt the index and query vectors, and meanwhile ensure accurate relevance score calculation between encrypted index and query vectors. 1) design a searchable encryption scheme that supports both

the accurate multi-keyword ranked search and flexible dynamic operation on document collection. 2) The special structure of our tree-based index, the search complexity of the proposed scheme is fundamentally kept to logarithmic. And in practice, the proposed scheme can achieve higher search efficiency by executing our "Greedy Depth-first Search" algorithm. Moreover, parallel search can be flexibly performed to further reduce the time cost of search process The search process of the UDMRS scheme is a recursive procedure upon the tree, named as "Greedy Depth- first Search (GDFS)" algorithm.

ManasiDoshi, SwapnajaHiray,[6] proposal describes that cloud storage allows users to store their data and enjoy high quality of services. It enables highly scalable, on demand and only pay per use services to be easily inspired over the Internet on an as needed base. Cloud stores data on remote machine, so necessary to need more security from unauthorized person. To achieve this, achieve flexible distributed storage, utilizing the homomorphism token and distributed erasure-coded data. Also allows for strong cloud storage correctness and simultaneously achieves fast data error localization. Security issue is very important in cloud there are many techniques available so here is review of all these. Data security is the major challenge in the cloud computing as user's data reside in the servers which are distantly positioned and far away from the end-users.

Jonathan Katz, Amit Sahai and Brent Waters,[7]Author propose that predicate encryption is a new paradigm for public-key encryption generalizing, among other things, identity-based encryption. In a predicate encryption scheme, secret keys correspond to predicates and cipher texts are associated with attributes; the secret key SK corresponding to a predicate f can be used to decrypt a cipher text associated with attribute I if and only if f(I) = 1. Constructions of such schemes are currently known for certain classes of predicates. Authors construct such a scheme for predicates equivalent to the valuation of inner products over ZN (for some large integer N).

Dan Boneh and Giovanni Di Crescenzo,[3]Here authors study the problem of searching on data that is encrypted using a public key system. Consider user Bob who sends email to user Alice encrypted under Alice's public key. An email gateway needs to test whether the email contain the keyword \urgent" so that it could route the email accordingly. Alice, on the other hand does not wish to give the gateway the ability to decrypt all her messages. Authors define and construct a mechanism that enables Alice to provide a key to the gateway that enable the gateway to test whether the word \burning" is a keyword in the email without knowledge anything else about the email. Authors refer to this mechanism as Public Key Encryption with keyword Search.

E.-J. Goh,[4]According to this work, a secure index is a data structure that allows a query with a "trapdoor" for a word x to test in O(1) time only if the index contains x; The index reveal no information about its contents without valid trapdoors, and trapdoors can only be generated with a secret key. Secure indexes are a natural extension of the problem of construct data structures with privacy guarantee such as those provided by oblivious and history autonomous data structures. In this work, Authors officially define a secure index and formulate a security model for indexes known as semantic security beside adaptive chosen keyword attack (ind-cka). Authors also develop a competent indcka secure index construction called z-idx using pseudo-random functions and Bloom filters, and show how to use z-idx to implement searches on encrypted data.

2.1 Problem Formulation

The problem of work is that document is not able to view and modify and there is no Rank method is applied for file to detect the number of times, whether the file is viewed or not. Secret key is also not generated in the exiting work. It does not allow for individual creation of password.

2.2 Existing Scenario

The proposed a secure tree-based search scheme over the encrypted cloud data, which supports multi-keyword ranked search and dynamic process on the document collection. Purposely, the vector space model with the widely-used "term frequency (TF) × inverse document frequency (IDF)" model are joint in the index construction and query generation to present multi-keyword ranked search. In order to find high search competence, Construct a tree-based index structure and propose a "Greedy Depth-first Search" algorithm based on this index tree. The secure kNN algorithm is utilized to encrypt the index and query vectors, and meanwhile ensure accurate bearing score calculation between encrypted index and query vectors. To resist different attacks in different threat models, build two secure search schemes: the basic dynamic multi-keyword ranked search (BDMRS) scheme in the known cipher text model, and the enhanced dynamic multi-keyword ranked search (EDMRS) scheme in the known background model.

2.3 Proposed Scenario

The data owner is dependable for generating updating information and sending them to the cloud server. Thus, the data owner needs to store the unencrypted index tree and the information that are required to recalculate the IDF values. Such an active data owner may not be very suitable for the cloud computing model. It could be a meaningful but hard future work to design a dynamic

searchable encryption scheme whose updating process can be completed by cloud server only, meanwhile reserve the aptitude to support multi-keyword ranked search. In adding, as the most of works about searchable encryption, our scheme mostly considers the challenge from the cloud server. Actually, there are many secure challenges in a multi-user scheme. Firstly, all the users usually keep the same secure key for trapdoor production in a symmetric SE scheme. In this case, the revocation of the user is big challenge. If it is needed to withdraw a user in this scheme, they need to rebuild the index and allocate the new secure keys to the authorized user. Secondly, symmetric SE schemes usually assume that all the data users are trustworthy. It is not sensible and a dishonest data user will lead to many secure problems. For example, a dishonest data user may search the documents and distribute the decrypted documents to the unauthorized ones. Even more, a corrupt data user may share out his/her secure keys to the unauthorized ones. The works, try to improve the SE scheme to handle these challenge problems.

Vector Space Model

The vector space model process can be divided in to three stages. The first stage is the document indexing where content behavior terms are extracted from the document text. The second stage is the weighting of the indexed terms to improve recovery of document relevant to the user. The last stage ranks the document with respect to the query according to a similarity measure. The vector space model has been criticized for being ad hoc. Vector space model along with TF×IDF rule is widely used in plaintext information retrieval, which efficiently supports ranked multi-keyword search Here, the term frequency (TF) is the number of times a given term (keyword) appears within a document, and the inverse document frequency (IDF) is obtained through dividing the cardinality of document collection by the number of documents containing the keyword.

- N_{f,w_i} – The number of keyword w_i in document f.
- N – The total number of documents.
- N_{w_i} – The number of documents that contain keyword w_i.
- TF'_{f,w_i} – The TF value of w_i in document f.
- IDF'_{w_i} – The IDF value of w_i in document collection.
- TF_{u,w_i} – The normalized TF value of keyword w_i stored in index vector Du.
- IDF_{w_i} – The normalized IDF value of keyword w_i in document collection.

The relevance evaluation function is defined as:

$$RScore(D_u,.Q) = D_u.Q = \sum_{w_i \in W_q} TF_{u,w_i} * IDF_{w_i}$$

If u is an internal node of the tree, TF_{u,w_i} is calculated from index vectors in the child nodes of u. If the u is a leaf node, TF_{u,w_i} is calculated as:

$$IDF_{w_i} = \frac{IDF'_{w_i}}{\sqrt{\sum_{w_i \in W_q} (IDF'_{w_i})^2}}$$

where TF'_{f,w_i} = 1+ ln N_{f,w_i}. And in the search vector Q, IDF_{w_i} is calculated as:

$$IDF_{w_i} = \frac{IDF'_{w_i}}{\sqrt{\sum_{w_i \in W_q} (IDF'_{w_i})^2}}$$

where IDF'_{w_i} = ln (1 + N/ N_{w_i}).

Advantages

- The secret key for the user is individually separated. If the one user founded guilty means they can't affect any of the users in the system.
- When the user wants to change their password the secret key also been updated. So, that the user's privacy more increased.

3. SYSTEM METHODOLOGY

- The search process of the UDMRS scheme is a recursive method upon the tree, named as "Greedy Depth first Search (GDFS)" algorithm.
- Based on the UDMRS scheme, build the basic dynamic multi-keyword ranked search (BDMRS) scheme with using the secure kNN algorithm.
- The BDMRS scheme can defend the Index Confidentiality and Query Confidentiality in the identified cipher text model.

3.1 System Architecture

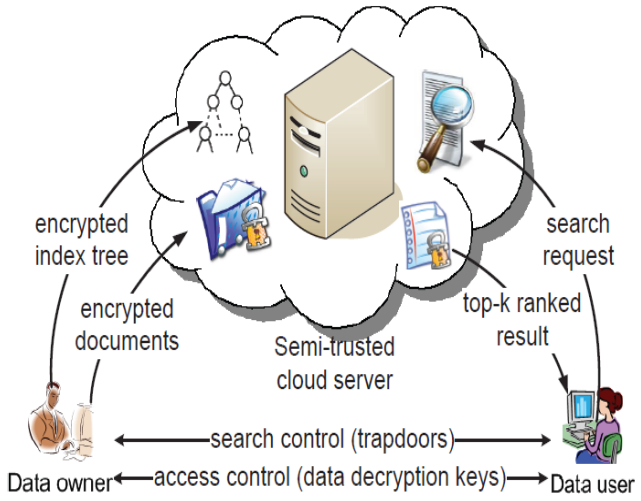

Fig-1 System Architecture

3.2 Module Description

- Search Process of UDMRS Scheme
- BDMRS Scheme
- EDMRS Scheme
- Individual Key and Password Update

Search Process of UDMRS scheme

The search process of the UDMRS scheme is a recursive process upon the tree, named as "Greedy Depth first Search (GDFS)" algorithm. Construct a result list denoted as *RList*, whose element is defined as *(RScore; FID)*. Here, the *RScore*is the relevance score of the document *fFID*to the query, which is calculated according to Formula

(1). The *RList*stores the *k* accessed documents with the largest relevance scores to the query. The elements of the list are ranked in downward order according to the *RScore*, and will be updated timely through the search process.

2). RScore(*Du;Q*) – The function to calculate the application score for query vector *Q* and index vector *Du* stored in node *u*.

kthscore– The smallest relevance score in accessible *RList*, which is initialized as 0.

hchild– The child node of a tree node with upper relevance score.

lchild– The child node of a tree node with lower bearing score.

Since the possible largest bearing score of documents rooted by the node *u* can be predict, only a part of the nodes in the tree are access through the search process.

Algorithm 1

Input: the document collection F = {f1,f2,.., fn} with the identifiers FID = {FID|FID = 1,2,..,n}.
Output: the index tree T

for each document f_{FID}in F do
Construct a leaf node u for f_{FID}, with u.ID =GenID(), u.Pl = u.Pr = null, u.FID = FID, and
D[i] = TF_{f_{FID},w_i} for i = 1,.....m;
Insert u to CurrentNodeSet;
end for
while the number of nodes in CurrentNodeSet is larger than 1
do
if the number of nodes in CurrentNodeSet is even, i.e. 2h then
for each pair of nodes u′ and u′′ in CurrentNodeSet
do
Generate a parent node u for u′ and u′′, with u.ID = GenID(), u.Pl = u′, u.Pr = u′′, u.FID =0 and D[i] = max{u′.D[i]; u′′.D[i]} for each i = 1,.....,m;
Insert u to TempNodeSet;
end for
else
for each pair of nodes u′ and u′′ of the former (2h − 2) nodes in CurrentNodeSet
do
Generate a parent node u for u′ and u′′;
Insert u to TempNodeSet;
end for
Create a parent node u1 for the (2h − 1)-th and 2h-th node, and then create a parent node u for u1 and the (2h + 1)-th node;
Insert u to TempNodeSet;
end if
Replace CurrentNodeSet with TempNodeSet and then clear TempNodeSet;
end while
return the only node left in CurrentNodeSet, namely, the root of index tree T ;

Algorithm 2 GDFS (index tree node *u*)

1: **if** the node u is not a leaf node **then**
2: **if** RScore(Du,Q) >k^{th}score**then**
3: GDFS(u:hchild);
4: GDFS(u:lchild);
5: **else**
6: **return**
7: **end if**
8: **else**
9: **if** RScore(Du,Q) >k^{th}score**then**
10: Delete the element with the smallest relevancescore from RList;

11: Insert a new element RScore(Du,Q); u.FID⟩andsort all the elements of RList;
12: **end if**
13: **return**
14: **end if**

BDMRS Scheme

The UDMRS scheme based on construct the basic dynamic multi-keyword ranked search (BDMRS) scheme by using the secure kNN algorithm. The BDMRS scheme is designed to achieve the goal of privacy preserving in the known cipher text model, and the four algorithms included are described as follows:

SK ← Setup() Initially, the data owner generates the secret key set *SK*, including

1) a randomly generate *m*-bit vector *S* where *m* is equal to the cardinality of vocabulary, 2) two (*m*×*m*) invertible matrices *M*1 and *M*2. Namely, *SK* = *{S;M1;M2}*.

I ← GenIndex(*F; SK*) First, the unencrypted index tree *T* is built on *F* by using *T* ← Build Index Tree(*F*). Secondly, the data owner generate two random vectors *{Du′;Du′′}* for index vector *Du* in each node *u*, according to the secret vector *S*. specially, if *S*[*i*] = 0, *Du′*[*i*] and *Du′′*[*i*] will be set equal to *Du*[*i*]; if *S*[*i*] = 1, *Du′*[*i*] and *Du′′*[*i*] will be set as two random values whose sum equals to *Du*[*i*]. Finally, the encrypted index tree *I* is build where the node *u* supplies two encrypted index vectors *Iu*= *{MT*1 *Du′;MT*2*Du′′}*.

TD ← GenTrapdoor(*Wq; SK*) With keyword set *Wq*, the unencrypted query vector *Q* with length of *m* is generated. If *wi*∈ *Wq*, *Q*[*i*] stores the normalize *IDF* value of *wi*; else *Q*[*i*] is set to 0. Similarly, the query vector *Q* is split into two random vectors *Q′* and *Q′′*. The disparity is that if *S*[*i*] = 0, *Q′*[*i*] and *Q′′*[*i*] are set to two accidental values whose sum equals to *Q*[*i*]; else *Q′*[*i*] and *Q′′*[*i*] are set as the same as *Q*[*i*]. Finally, the algorithm returns the trapdoor
TD = *{M*−11 *Q′;M*−12 *Q′′}*.

Relevance Score ← SRScore(*Iu;TD*) With the trapdoor *TD*, the cloud server computes the significance score of node *u* in the index tree *I* to the query. Note that the relevance score calculated from encrypted vectors is equal to that from unencrypted vectors.

EDMRS Scheme

The BDMRS scheme can defend the *Index Confidentiality and Query Confidentiality* in the known cipher text model. However, the cloud server is able to link the similar search requests by track path of visited nodes. In addition, in the known backdrop model, it is possible for the cloud server to recognize a keyword as the normalize TF division of the keyword can be exactly obtained from the final calculated significance scores. The primary cause is that the relevance score calculated from *Iu* and *TD* is exactly equal to that from *Du* and *Q*. A heuristic method to further improve the security is to break such exact equality. Thus introduce some tunable arbitrariness to upset the relevance score calculation. In addition, to ensemble different users' preference for higher accurate ranked results or better sheltered keyword privacy, the randomness is set adjustable.

Individual key and Password Update

The individual key and password update scheme can protect the data from the unrecognized user. The user has to submit their given secret key values at the time of login. So, the data in the cloud is more protected now then the previous scheme. If user feels their password has been leaked then they can update their password in the cloud. When the user wants to change their password the secret key also changed. So, the protection of the cloud environment is more increased. The most important aspect of our enhancement is data owner's description about the file which they are uploading to cloud. This description content reduces the time to build the search index and searching time of the system.

4. RESULTS AND DISCUSSION

Trapdoor Generation

The generation of a trapdoor incur a vector split operation and two multiplications of a (*m* × *m*) matrix, thus the time complexity is *O*(*m*2).

Time for Search

The time taken to search the documents that are related to the documents in the BDRMS scheme is more complicated then the IKP-BDRMS scheme. In the IKP-BDRMS scheme the data owner provide the keyword related to the document which they are uploading it. So, the searching time of the document considerably reduced.

Precision Test

The search precision of scheme is precious by the dummy keywords in EDMRS scheme. Here, the 'precision' is defined as *Pk*= *k′*=*k*, where *k′* is the number of real top-*k* documents in the retrieved *k* documents. If a smaller standard deviation is set for the random variable Σ″*v*, the IKP format is supposed to find higher precision, and vice versa.

Table-1: Precision Test

Method	Precision
IKP	87
BDMRS	93

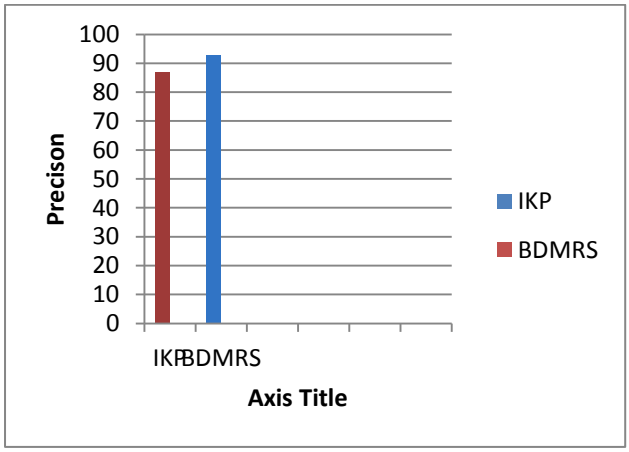

Chart 1-Precision Test

5. CONCLUSION

Search retrieval is the most essential task in the cloud based search engines where the multi cloud users attempts to retrieve the contents based on their need. In this case security and privacy becomes the most important issue where the users need to submit their information to retrieve the result as per their requirements. In this work, privacy preserved search retrieval based on multi keyword search is introduced which will retrieve the contents in terms of user submitted query in the Vector space model. Along with these, the individual key and Password update based security scheme is supported to enable the multi keyword search retrieval in the more secured manner. The experimental tests conducted were proves that the proposed approach provides better result than the existing work in terms of improves search retrieval accuracy and the privacy. In future work, it can include role based access concept to our system. By applying role based concept, it authorizes the user who can access and what they access based on their roles.

REFERENCES

[1]. Boneh .D, Di Crescenzo .G, Ostrovsky .R, and Persiano .G, "Public key encryption with keyword search," in Advances in Cryptology- Eurocrypt 2004. Springer, 2004.
[2].Boneh .D, Kushilevitz .E, R. Ostrovsky .R, and Skeith .W.E III, "Public key encryption that allows pir queries," in Advances in Cryptology-CRYPTO 2007. Springer, 2007.
[3].DanBoneh and Giovanni Di Crescenzo" Public Key Encryption with keyword Search"Volume 3027 2004 pp 506-522.
[4].Goh .E.-J et al., "Secure indexes." IACR Cryptology ePrint Archive, 2003.
[5].Jonathan Katz, Amit Sahai and Brent Waters" Functional Encryption: Definitions and Challenges"2006.
[6].ManasiDoshi, SwapnajaHiray" Secure and Data Dynamics Storage Services on Cloud" Volume 3,2013.
[7]. Wang .C, Ren .K, Yu .S, and Urs .K.M.R, "Achieving usable and privacy-assured similarity search over outsourced cloud data," in INFOCOM, 2012 Proceedings IEEE. 2012.
[8]. Wenhai Sun, Wenjing Lou, Y. Thomas Hou, and Hui Li" Verifiable Privacy-Preserving . ....Multi-Keyword Text Search in the Cloud Supporting Similarity-Based Ranking". 2014, Volume: 25.
[9]. Yakoob. Sk., Dr. V Krishna Reddy and C. Dastagiraiah "Analysis of Keyword Searchable Methodologies in Encrypted Cloud Data".
[10]. Zhihua Xia, Xinhui Wang, Xingming Sun and Qian Wang "A Secure and Dynamic Multi- keyword Ranked Search Scheme over Encrypted Cloud Data" IEEE Transactions 2015.